LEGO ROCK RAIDERS

Produced by Data Design Interactive c 1999

Level Objectives using NERPS

Example 1:
Find four trapped men to complete the level.
Lose your toolstore and you fail the mission

Nerp file

```
GetObjectiveShowing != 1 ? :Skip
#Set number of hidden men found
TRUE ? SetR7 0
#Clear Mission complete flag
TRUE ? SetR6 0
Stop
Skip:
GetToolstoresBuilt = 0 ? SetLevelFail
GetHiddenObjectsFound = GetR7 ? :NoneFound
TRUE ? SetR7 GetHiddenObjectsFound
TRUE ? SetMessage GetR7 0
NoneFound:
GetR6 > 0 ? :CloseDown
GetR7 = 4 ? SetR6 1
TRUE ? SetTimer1 0
Stop
CloseDown:
GetTimer1 > 5000 ? SetLevelCompleted
Stop
```

Message File

```
Well done - You have freed a trapped Rock Raider
Fantastic - You have freed another trapped Rock Raider
Amazing - You have freed yet another Rock Raider, only one more to go
Incredible - You have freed all the trapped Rock Raiders
```

Breakdown

When the game first starts the objective is showing. The NERPS detects this using **GetObjectiveShowing**. Initially the code sets both R7 and R6 to zero. R6 is used as a flag for the mission being complete. R7 is used as a counter for the number of men discovered. These registers will be zet to zero as long as the objective is showing. As soon as the space bar is pressed, the objective will no longer be showing, and the code will jump to the **:Skip** label.

The code now checks to see if there are any toolstores built using the **GetToolstoresBuilt** command. If this function returns zero then the toolstore is lost. The code straight way calls **SetLevelFail** which tell the game that the mission was a failure.

If there are toolstores remaining, the code continues to check the number of hidden objects found using **GetHiddenObjectsFound.** This function will return zero initially. This is the value in R7 to start with. The code will find **GetHiddenObjectsFound** and R7 equal and jump to the **NoneFound:** label. More on what happens here later

If **GetHiddenObjectsFound** and R7 are not equal at any point, this means that additional hidden objects have been found. This NERPS program flags this to the player by printing a message to the display. The command for this is **SetMessage** which takes two parameters. The first is the line number of the text file to print. The second is a flag to determine whether to press an icon to continue past the message display. Zero will choose to enable the press an icon to continue display. As you discover hidden Rock Raiders **GetHiddenObjectsFound** will return values from 1 to 4. This will be passed into R7 and then into **SetMessage** which will display the relevant text message on screen. See the message file listing above. Each of the four lines will be displayed, if the player is successful. Since R7 is set to the value of **GetHiddenObjectsFound** , **SetMessage** will not be called again until another hidden object is found.

We have now reached the **NoneFound:** label. The function of the following code is to add a small time delay after the objective has been completed. If R6 is non-zero then we are in the countdown stage. The objective has been achieved and wwe are waiting the allotted time to elapse. The code will jump to CloseDown: which checks timer 1 to see if 5 seconds (5000 milliseconds) have elapsed since being set to zero. When this happens, **SetLevelCompleted** is called which tell the game to end, with the level completed successfully.

If we are not in the countdown stage, then the code falls through to checking R7 to see if the number of hidden Rock Raiders found has reached four, the target. When this happens, R6 is set to one, and timer 1 is initialised to zero, to enable the countdown stage.

Well that's it, simple really. I don't know what all the fuss is about

Example 2:
Find and store 20 crystals to complete the level.
Lose your toolstore and you fail the mission

Nerp file

```
GetObjectiveShowing != 1 ? :Skip
#Set number of crystals found and stored
TRUE ? SetR7 20
#Clear Mission complete flag
TRUE ? SetR6 0
Stop
Skip:
GetToolstoresBuilt = 0 ? SetLevelFail
```

```
GetCrystalsCurrentlyStored = GetR7 ? :NoneStored
TRUE ? SetR7 GetCrystalsCurrentlyStored
TRUE ? SetMessage GetR7 0
NoneStored:
GetR6 > 0 ? :CloseDown
GetR7 = 20 ? SetR6 1
TRUE ? SetTimer1 0
Stop
CloseDown:
GetTimer1 > 5000 ? SetLevelCompleted
Stop
```

Message File

```
You have no crystals stored
You have stored one crystal
You have stored two crystals
You have stored three crystals
You have stored four crystals
You have stored five crystals
You have stored six crystals
You have stored seven crystals
You have stored eight crystals
You have stored nine crystals
You have stored ten crystals
You have eleven crystals stored
You have twelve crystals stored
You have thirteen crystals stored
You have fourteen crystals stored
You have fifteen crystals stored
You have sixteen crystals stored
You have seventeen crystals stored
You have eighteen crystals stored
You have nineteen crystals stored
You have stored twenty crystals, objective achieved
You have stored twenty one crystals, objective achieved
```

Breakdown

You'll find this very similar to example 1. The first stage initialises the relevant variables. R6 is the objective achieved countdown flag. R7 is the count of crystals collected. The only difference is that instead of checking for the number of hidden objects found, we are now checking for the number of crystals found and stored. Everything else functions in a similar manner. The text file may be repetitive, but is effective..

Compiling your NERPS file

- Copy NERPC.EXE to your c:\windows directory, or any pathed directory

- NERPC is a DOS based executable. Use under the DOS environment. (DOS prompt etc.)
- NERPS source files usually have the .NRN extension. The NERPS executable usually have the .NPL extension
- Make sure if you are compiling over a .NPL file, that the file is not read only. The .NPL will not be changed, and no error will be reported. Files may be read only if they have been copied from CD. The read only attribute may be removed by using **ATTRIB <filename> -r**
- If you have create L01.nrn in the Lava01 directory. Under DOS, go to the relevant directory. Create the NERPS executable by typing **NERPC L01.nrn L01.npl.** L01.npl will be created for use in the game
- Any NERPS related text files must be created without any kind of formatting. Notepad is quite safe to use. Wordpad and Word may format your text, and create content which may confuse the NERPS compiler

Adding NERPS files to levels

You have lovingly created L01.npl and L01.txt files. You wish to add these files to the Lava01 level. In LEGO.CFG, go to the level section to which you wish to add the NERPS file, and the associated message file and add the following.

```
NERPFile            Levels\GameLevels\Lava01\L01.npl
NERPMessageFile     Levels\GameLevels\Lava01\L01.txt
```

Job done